Executive Overview:

Migrating Applications to NMCI: Web-/Portal-Enablement; New Tools

WRQ is a member of the ISF and is the author of Reflection which is a gold disk application for NMCI. In addition to our Reflection tool, we also offer a programmatic integration capability, Verastream, which aids in the migration of legacy applications to the web through the creation of web services from those legacy environments currently supported by Reflection. WRQ Verastream® software encapsulates host functionality into services for rapid reuse in web applications, packaged applications, and portals. Gartner ranks WRQ in the leadership quadrant for programmatic integration.

We are confident the web services created by Verastream provide an excellent migration tool for use within the WEN framework. Verastream is also under consideration by the Marine Corp in support of their MCEITS initiative as a migration tool. It is also our opinion the web service components created by Verastream are also capable of supporting the NCES framework for the Department of Defense which provides an infrastructure to facilitate cross domain information and code sharing to maximize intelligent operation and minimize cost expenditures. Verastream is a functional tool that specifically would speak to the data mediation needs for NCES.

The attached white papers speak to:

- Methods of receiving demonstrable value from Web Services
- The value of a Service Oriented Architecture through the use of a Service Oriented Integration tool creating web services
- Increasing the value of Portal solutions thru the use of legacy “portlets”
- Tangible ROI thru Customer examples: Raytheon, State of Minnesota, Colorado Student Loan, AT&T, Navy Perscom- Reserve Payroll Analysis
- Summary: MOVING LEGACY APPS TO THE WEB- A Time-Saving Process, A Resource-Efficient Approach

How to Get Real Value from Web Services ...
Strategies for Building Secure Web Services From Mainframe Transactions

With the foundational Web services technologies--XML, SOAP, WSDL--being widely adopted by major platform and application vendors, Web services have become a practical way for organizations to reduce the cost of application integration and align IT more readily with the dynamic nature of business demands. Certainly there are gaps in the functionality provided by the basic Web services standards--specifically in the areas of security, transactional integrity, quality of service and management--yet companies are forging ahead with Web services projects.

The widespread belief is that the technologies are here to stay. And the benefits of the technologies, when used in the right circumstances, are significant enough to overcome concerns about the missing functionality.

Deciding Where and When to Use Web Services

Although Web services are made up of new Internet-based standards, what they are trying to achieve is not new. They build on previous models for distributed computing and make the concept of software reuse more achievable. Just as Web services are an extension of previous computing models, their use must make sense in the context of existing business models.

A good place to start with Web services is to find the best reuse scenario, one that meets the organization's demands for information sharing and has a demonstrable ROI. For example, businesses can begin by automating manual processes where information is rekeyed into disparate applications. In this case, Web services could speed up processing time and reduce data entry errors. This effort would require modeling the manual business process, figuring out which back-office systems are involved, enabling those systems with Web services interfaces and then integrating them.

Replacing batch cycles with real-time processing is also possible in the Web services world. This makes sense if zero latency is important or if existing applications are running out of batch cycles. It may be workable to replace expensive EDI transactions with Web services or to add automated information exchange with partners that never embraced EDI. Web services are an easier, less-expensive way to share data among trusted business partners.

Syndicating services to partners provides a payback in many industries. For instance, insurance companies may want to syndicate claims inquiries to their agents. Wrapping their existing claims inquiry functionality as a Web service makes it easy for agents to embed the service in their own Web pages.

Sharing information across multiple lines of business or within merged businesses is another scenario where Web services are already providing value. In many instances, a company's customer information is scattered across multiple systems and applications. Consolidating all this customer information into a single view is a goal shared by many companies. Web services interfaces on existing applications make this effort easier, faster, less expensive and more adaptable than moving around the customer data.

Bringing all kinds of legacy applications into the newer Web environment is easier to do because of Web services. Putting these applications into a query/response situation is now possible. Once legacy applications are enabled with Web services interfaces, they can be reused easily, and the information they store can be exposed to more people in less-expensive ways.

For those applications that aren't already componentized and exposed as Web services--the majority of applications--tools are available to non invasively create components of the right granularity for the task and put a service interface on them. Low-level components are created from adapters to various custom and packaged applications and then registered in a repository.

These low-level or atomic components encapsulate the details of the application so users of the component don't have to deal with the semantics of the low-level API.

From the repository, atomic components and others can be connected with routing logic and data transformation and assembled into high-level components. Once high-level components are constructed, they can be exposed as services for communicating with client applications. These services can then be registered for access in a new business process. A composite application is the deployment mechanism for the newly composed process.

Using components and Web services, for instance, a new application can be built in which customer order information from a CICS application is combined with product inventory information from an application running on an AS/400 and returned to the customer via a Web interface. The application combines proven functionality in new ways to benefit the business.

The Move to a Services-Oriented Architecture

Transitioning to a Web services development environment requires a shift in infrastructure and culture. IT infrastructure must move toward hosting components and services rather than monolithic applications, and developers must get accustomed to writing software as reusable components and building new applications by assembling available components and services that are useful to more than one application. This means implementing a services-oriented architecture (SOA) and thinking in terms of composite applications and services.

An SOA is an architecture where applications are decomposed into services and made available on the network. Rather than being built in the classic monolithic way, applications can be broken up into components and exposed as services without reengineering or rebuilding them, taking advantage of new state-of-the art technology.
The technologies used to provide the services to clients can be technologies such as JMS and SMTP in addition to the Web services technologies SOAP, WSDL, HTTP and XML. Not only does an SOA enable a great deal of flexibility in deployment, it also inherently delivers on the ability to scale as the enterprise grows. Since services must be exposed through a directory mechanism (in many cases UDDI) in order to be accessed, it is a simple matter to use that directory to distribute requests to locate and use multiple services as they are made available. Widely disparate systems can share data and logic.

In an SOA, process, presentation, business logic and data are all on separate tiers and are loosely coupled, allowing the rapid change of business processes. An organization can bring in new best-of-breed applications and integrate them easily with existing systems. An SOA promotes reuse, thus reducing the time it takes to deliver new business functionality.
With a services-oriented architecture, an organization can build incremental new functionality and couple it with existing functionality to form composite applications. Composite applications are new business applications constructed through the combination of existing data and business logic with new process flows and new business logic. Because they leverage existing, reliable resources within an enterprise, composite applications promote the rapid delivery of new business functionality. In addition, since composite applications reuse proven business logic, they are also more robust than applications built from scratch.

If an organization takes the composite application approach, all its existing systems can be looked at as reusable assets. Components and services within these systems can be defined for reuse and consumption by other systems.
Composite applications are not a new concept. When the idea was introduced in the early days of client/server computing, a composite application was something assembled from a number of components. Web services add another layer of abstraction to mask the underlying differences in technology. The mechanism that allows services to be registered, discovered, requested and

delivered, regardless of their system of origin, makes Web services considerably more flexible than a classic component model.

Service-oriented integration

Service-oriented integration is an emerging solution to the problem of extending existing information assets.

Traditional approaches have focused on two aspects of the overall problem:

• Message brokers and extract/transform/load (ETL) products have addressed the problem of synchronizing data across multiple applications.

• Business process management (BPM) products enable business to coordinate the actions of multiple independent applications.

There is an emerging requirement, however, to utilize functionality in existing applications to support new application development. This scenario differs from either of the above because it is focused on repurposing existing functionality for use in new applications. Service-oriented integration addresses this requirement by encapsulating functionality in existing applications and exposing it through standard interfaces. The resulting services can then be easily consumed by new applications.

The need to reuse existing functionality arises from several factors. Businesses are being asked to respond to changes in the market more quickly than ever before. New business models and partnerships form and dissolve in weeks and months, as opposed to years. These changes are driving the demand for new applications. At the same time, competition has reduced operating margins. Budgets, particularly for information technology, have stagnated or have been reduced. IT organizations are being asked to do more with less.

Businesses are also burdened with operational applications dating back to the 1970's. These applications, typically host-based, were highly customized to meet the needs of the organization at that time and lack modern interfaces. These applications present a serious challenge to IT organizations needing to adapt to new user communities and access methods, such as the Internet.

Services Encapsulate Existing Functionality

Service-oriented integration addresses these problems by enabling IT organizations to offer the functionality in existing applications as reusable services. Simply put, a service is a published interface, providing transparent access to functionality over a network.

Some of the key benefits of services are:

• Well-defined interface — applications written to consume services are assured of easily-understood and consistent access to the underlying functionality

• Opaqueness — the technology and location of the application providing the functionality is hidden behind the service interface

• Flexibility — both providers of services and consumers of services can change. As long as they continue to adhere to the interface, the applications will continue to work

For example, an insurance company might offer a service to enable agents, third-party providers, and customers to check on the status of their claims.
The claim system is currently a COBOL/VSAM application, running on an OS/390 host. The insurer encapsulates the claim status check functionality as
a service. Applications can now be developed that utilize this service, independent of the claim system. These consuming applications are completely unaware of the technology underlying the service.

In fact, the insurer may migrate to a new claim system in the future. As long as a claim status check service that adheres to the original interface is
available, the consuming applications will continue to function. A single service can expose a variety of interfaces, including SOAP, HTTP, COM, and Java.

Having multiple interfaces enables the service to be called from a wide range of applications. The result is a flexible, scalable approach to integration. Encapsulating existing application functionality as services enables IT organizations to respond to new demands quickly. Applications that
provide services also can be migrated to new platforms or technologies, without affecting the applications that consume these services. Services
can be a foundation for a long-term application migration strategy.

Multi-level Encapsulation Enables More Useful Services

Another key aspect of service-oriented integration is the ability to encapsulate application functionality at multiple levels. Enterprise applications are frequently complex, requiring specialized knowledge to
understand their operation. This knowledge can be built into a set of services, which can then present a simplified interface to the application. A consumer
of these services no longer requires detailed knowledge of the application to access its functionality.

Host applications, in particular, may have particular conventions or dependencies that are well known to long-time users of the application.
Rather than requiring this same level of knowledge throughout the organization, the conventions are hidden behind the service interface. The result is a service that can be consumed by users unfamiliar with the underlying applications.

This concept of encapsulation can be extended to higher levels as well. Frequently, established processes within a business require coordinated
actions across multiple applications. For example, establishing a new customer account may require creation of records within a CRM system as well as an order management application. Inter-application dependencies like these can be encapsulated into a higher-level service, which calls on lower-level services to access individual applications. The higher level service also coordinates actions across multiple applications, encapsulating the organization-specific interrelationships into a single service.

Services Allow for Incremental Development

The reusability and flexibility of services enable IT organizations to approach integration problems from a project perspective. Rather than deploying a costly and complex infrastructure prior to addressing any integration problems, service-oriented integration allows problems to be addressed one at a time.

Services can be created to fulfill a specific integration requirement. Later, these same services might be reused to fulfill a new requirement. In this way, IT organizations can create a library of reusable services over time. Eventually, new integration requirements might be fulfilled entirely through existing services.

Service-oriented integration is a flexible approach that enables IT organizations to leverage functionality easily in existing applications.

Developers can create services quickly, and reuse these services across multiple projects. As a result, businesses can realize initial benefits faster, and see those benefits grow over time. The ability to encapsulate functionality at multiple levels creates more useful services that can be reused across a wide range of new applications. Finally, the opaqueness of services enables businesses to grow and change over time. Existing applications can be migrated to new technology platforms, or even replaced, without affecting new applications that rely on their functionality. As a result of implementing SOI, organizations can quickly respond to change and continue to grow over time.

FILLING YOUR PORTAL TO MAKE IT PRODUCTIVE

Legacy Portlets Deliver Quick Results

Is your portal providing employees, customers, and partners with consolidated access to the information they need? Or is it left unused because it's empty, and just doesn't offer enough value? In order for portals to be productive, they must afford ready access to key business processes, applications, and information from around the enterprise.

INDUSTRY SOLUTIONS
The problem is that more than 70 percent of the necessary corporate functionality and information is locked up in legacy-application silos throughout the organization. So the first thing you need is a way to expose the logic and data that resides in these applications.
Then you can develop legacy portlets, which will be the key to filling your portal and making it a productive, unified asset for your organization.

Sizing-Up Your Portal Users
Effectively filling an empty portal requires some upfront evaluation of intended users' skills and needs. You probably have different types of users who need different levels of personalized access to host applications through the portlets you develop.

Power users, for example, need the full functionality of the green-screen application available within their portlet. They are comfortable with the nuances of the host, because they spend much of their day using it.

For these users, the current navigation, workflow, and other functionality of the host application can be offered through the portlet.
Casual or new users need legacy information available within their portlet, but they are not familiar with the complex navigation and workflow of most green screen applications. For these users, it is important to capture key business logic and data for simplified reuse within the portal framework.

Matching Functionality to User:

Portlet Customization
WRQ® software offers a single solution for many levels of access to host-application functions, workflow, logic, and data. You can easily develop portlets that give your users as little or as much functionality as they need to be productive.

Let's use the example of a host application focused on inventory and order processing to show how the transformation takes place.

Following are some WRQ options for creating, customizing, and publishing the legacy portlets that make it possible. Keep in mind that once the host data and logic are exposed, they can be reused and presented in various portlets to meet user needs.

Green-screen portlets
Suppose you want to set up a portlet for your order entry workers, who are very familiar with the host application, including its navigation and workflow. Green-screen portlets provide access to the underlying host application in its current state. You simply encapsulate the application "as is," resulting in a portlet that is well-suited to these power users. They get the full functionality of the host application, including file transfer and printing, while at the same time being governed by the portal access controls. To expand the reach of legacy information, consolidate host applications into easy-to-use portlets.

" At Schenker, WRQ improved both customer and employee satisfaction by making seven mission-critical legacy applications accessible through their portal. You can't underestimate the value of that for any company."

— KJELL JENNSTIG
Cap Gemini Ernst & Young Group

Most IT organizations want to use a combination of these low-risk, low-cost approaches. And as your portal grows, you'll need a host-application integration solution that enables you to update and evolve legacy portlets over time to meet new business needs.

WRQ has been helping organizations make the connection between host applications and emerging technologies for more than two decades. When PCs started changing the world of business in the early '80s, WRQ created terminal emulation software to deliver host access on a single desktop. More business changes took place with the subsequent arrival of the web, packaged applications, and portals. To meet a new set of IT challenges, WRQ tapped our host expertise and introduced advanced software for host application integration. Now, with paradigms such as SOA shifting the IT landscape yet again, WRQ solutions are keeping pace so you can continue getting the most from your host.

Working in the portal environment gives the order entry staff easy access to other information they need, so they can be more productive.

With WRQ software, users can launch a terminal emulator as a portlet, so they get the green-screen application in its entirety. Access controls ensure that users connect to the appropriate host application, based on who they are. The user menus and preferences in the emulation software can be applied to the emulation session in the portlet.

ASP- and JSP-based portlets

You might have customer-service representatives who need a simplified way to view up-to-date inventory, so they can give customers accurate information. The ideal portlet for these CSRs would provide a web GUI with certain host screens and fields eliminated, so they can focus on their tasks.

An ASP- or JSP-based portlet relies on the underlying host application’s workflow, but modernizes the user interface. It lets you deploy original or specialized legacy functionality to existing and new CSRs alike. Benefits include reduced training costs and improved customer service.

WRQ software supports automatic generation of JSP or ASP web applications from green-screen applications. Using on-the-fly HTML production, developers can integrate and customize any number of screens to tailor the interface to user needs. Web services as portlets

In another scenario, a Logistics command might need a dashboard-type portlet to track daily inventory and orders processed. This executive does not want to deal with complex navigation or extraneous details embedded in host screens. For this business need, web services are the ideal technology. They allow you to create a portlet that relies more on business logic and data than host-application workflow. (You can also use web services to combine multiple host applications for an entirely new purpose—say, a customer self service portal.)

WRQ software can abstract host functionality into reusable components with standard interfaces (including .NET, Java, COM, EJB, XML/HTTP, JMS, and web services). These components, or services, can be exposed directly within a portal page or combined with other services to form a composite application.

A service-oriented architecture (SOA) easily accommodates the composite-application portlets that result from this process.

Customer Examples:

Raytheon Missile & Space

"Verastream makes the mainframe act like any other database and provides excellent support for Microsoft .NET. This makes host integration fast and intuitive for our developers. WRQ delivered what they promised."

—Rob Vettor
Applications Architect
Raytheon

WHEN BUILDING THE MOST PRECISE RADAR SOLUTIONS ON EARTH,
TRACKING THE EXACT LOCATION OF EACH AND EVERY PART IS CRITICAL.
THAT'S RAPID HOST INTEGRATION. MADE POSSIBLE WITH WRQ VERASTREAM®.

In two months, Raytheon streamlined its materials handling process by exposing host functionality and creating a fast, easy-to-use Web-based materials tracking application. With Verastream, Raytheon developers can work within their .NET environment, manipulating information in the mainframe as if it's just another database. The result is real-time tracking, which helps Raytheon build pinpoint-accurate radar systems with pinpoint accuracy.

Results
• Saved approximately $200,000 per year

WRQ Verastream can do the same for you:
• Terminal and transactional host adapters
• Java and .NET support
• Visual integration tools
• Reusable services for multiple applications

Minnesota Dept. of Public Safety Driver and Vehicle Services Division

Minnesota Driver and Vehicle Services is an agency that serves a variety of constituents including car owners, drivers, dealers, and lessors; driver-license and law-enforcement agencies; and state, county, and city government units. DVS wanted to provide self-support to this broad spectrum of constituents and improve internal efficiencies for its 800 on-site and remote users. WRQ Verastream® Host Integrator helped them deploy a series of applications to meet these goals and quickly achieve measurable results.

PROBLEM

Difficult host information access for internal and external users

SOLUTION

Used Verastream Host Integrator to allow web self-support

RESULTS

- $260,000 saved in programming and legacy storage costs
- License processing times cut by 75 percent
- Improved functionality delivered in four weeks

Colorado Student Loan Program

Business needs

- Give financial aid administrators web-based, self-service access to student loan information

- Stay competitive with large guaranty agencies

Solution

- Used Verastream Host Integrator to automate manual processes and web enable host applications

Results

- Improved customer satisfaction with self-service web applications

- Retained guarantees for $390 million in student loans

AT&T Global Customer Care

Communications giant AT&T used Verastream Host Integrator to expedite billing, inquiry, and service calls from the hundreds of thousands of AT&T customers who have toll-free numbers.

Business need

- Reduce time and costs spent on customer calls by simplifying CSR access to information

System

- IBM MVS machines

Solution

- Verastream Host Integrator

Results

- Typical response time: previously seven to 10 minutes; now 30 seconds or less

• Savings of $750,000 per year in the cost of serving customers

• Dramatic increase in CSR productivity and caller satisfaction

Navy Perscom: Reserve Payroll challenge: TFW sponsored analysis

Business Problem

Multiple Legacy system interface challenges severely delay resolution of payroll issues.

Currently, when a payroll problem is reported a small team of individuals in Millington, TN is responsible to validate the request. The gathering of service information on military personal requires the accessing of several legacy systems and several components of these systems. They include: IMAPMIS Personal Data Collection, OPINS Personal Information, OPINS Present Service, OPINS Supplemental Gain Information, OPINS Various Dates, Designators, and Indicators, OPINS Miscellaneous Date Changes, Payroll Administration Data, Attendance Calendar (if reserve) and NEOPS Member's Anniversary Summary and Detail Records. This information is printed and a 25 – 40 page Statement of Service document then has to be manually reviewed to validate the change request.

Assessment of potential Verastream Solution for Pilot

Integrate these systems using Verastream so that they can be read from an web based HTML/ASP user interface.

- Legacy Business process captured within Verastream aiding in transitioning to People Soft
- This UI would present all the data required to validate the request on one screen.
- This would eliminate the need to logon to each of these systems manually, print the required pages and then assemble for review.
- Result in a re-engineered, Web service based composite application
- Project completion estimated in 4 weeks, dramatic reduction of resolution time anticipated

MOVING LEGACY APPS TO THE WEB- A Time-Saving Process, A Resource-Efficient Approach

Verastream simplifies the creation of web applications that are based on legacy functionality. Advanced technology gives you the ability to unlock your legacy systems and abstract the business functions so you can make them available for new purposes. And you can use the technical skills that you have on your staff right now.

ACCESS.
INTEGRATE.
TRANSFORM.™

There is an escalating demand to make core business functions available on the web—ASAP. That presents some tough challenges for Navy IT organizations already busy executing other initiatives. Core business functions, locked up in legacy systems, are not easily accessed or changed; webifying them can be a painstaking and labor-intensive process. What's more, your IT staff's expertise might not be well-suited to the job, and there's no time to acquire new skill sets. These hurdles, taken separately or together, make it impossible to establish any momentum in moving your legacy assets to the web.

Verastream accelerates the web app development process with automatic generation of web pages, JavaBeans,and web services. It also simplifies the collaborative development effort between integration developers and web developers; special tools, process automation, and open standards mean that they can leverage their own existing skill set while jointly developing a legacy-based web application.

How Does it Work?

Here are Verastream's three quick steps to legacy/web integration:

Step 1: Model the host application

Who: Integration developer

How: Verastream Design Tool

The graphical Design Tool encapsulates legacy logic, including screen navigation, without changes to the host. It lets integration developers create host application models to abstract legacy functionality to any level needed for the new application.

Time: Minutes to days, depending on the level of abstraction Alternative method: Integration developers spend weeks learning how to communicate with the host and navigate it programmatically.

Step 2: Generate the web pages

Who: Integration developer

How: Verastream Web Builder

Web Builder automatically produces web pages (regardless of the level of modeling that's been done) to serve as a basis for the new web application. These need different abstraction levels for different audiences?

Verastream offers three levels of abstraction for bringing legacy applications to the web. Depending on user needs, you can choose a no-modeling, light-modeling, or component development approach:

• No Modeling
You leave the host screen "as is," with no abstraction. In less than five minutes, you can get an unlimited number of host screens automatically represented as new web pages. This level of abstraction is appropriate only for users who are already familiar with the legacy application.

• Light modeling

In two or three minutes per screen, you can hide unnecessary fields and add more intuitive navigation buttons. Even though the screens are customized at this stage, Verastream instantaneously auto-generates web pages from them. This level of abstraction helps you extend your application to new users, as long as they are familiar with your business processes.

- Component development

In just one to two days per function, you can encapsulate legacy functionality—spanning multiple host screens—into a single component. JavaBeans or web services can be auto generated from the component, which in turn can be used to build new web applications. The resulting web application will not be limited by the appearance or the flow of the legacy application. This level of abstraction lets you build in the right functionality for the newest users, including those who have had no exposure to your business processes.

For optimum flexibility, you can mix and match these processes. Use a combination of no modeling + light modeling, or light modeling + component development, to get just the result you need.

Step 3: Design the web application

Who: Web developer

How: Developer's preferred JSP-,Java Bean-, or web service supported design tool or IDE

Web developers can concentrate on design, rather than on establishing links between webpage functions and legacy-app functions. In addition, the code that links the functionality between systems has already been tested.

Time: Minutes to days, depending on
how much they want to design

Alternative method: Web developers spend weeks learning new tools or IDEs.

With Verastream, you get a practical way to share information between your legacy and web systems, while leaving your back-end legacy logic